Understand and use essential tools

Access a shell prompt and issue commands with correct syntax

. start bash	$ bash
. install bash completion (needs reboot)	$ yum install bash-completion

Use input-output redirection (>, >>, |, 2>, etc.)

. overwrite dump.log with last 25 lines from messages

$ tail -n 25 /var/log/messages > dump.log

. Append last 25 lines from messages to dump.log

$ tail -n 25 /var/log/messages >> dump.log

. Redirect the standard errors to error.log

$ grep test /proc/* 2> error.log

Use grep and regular expressions to analyze text

. Show current sessions with a grep on ssh	$ ps -aux | grep ssh
. Show all network related log messages:	$ cat /var/log/messages | grep network
. show first 5 lines of a file	$ head -n 5 </file>
. show last 5 lines of a file	$ tail -n 5 </file>
. show the 5th line in a file	$ head -n 5 </file> | tail -n 1
. get regex manual and examples	$ man 7 regex

Access remote systems using SSH

. Most basic remote ssh session setup	$ ssh <username>@<remote-host>
. Use SSH keys	$ ssh-keygen
. use the defaults, no passphrase	[]enter
. distribute the public key (id_rsa.pub)	$ ssh-copy-id <hostname-or-ip>

Log in and switch users in multiuser targets

. Show information regarding current user account	$ id
. Switch the current user to another user	$ su - <user-name>
. Switch the current user to Root	$ su -

Archive, compress, unpack, and uncompress files using tar, star, gzip, and bzip2

. Create archive	$ tar -cf <new-archive.tar> </dir>
. Create gzip archive	$ tar -czf <new-archive.tar.gz> </dir>
. Create bzip archive	$ tar -cjf <new-archive.tar.gz> </dir>

. Extract tar archive	$ tar -xf <archive.tar>
. Extract tar gzip archive	$ tar -xzf <archive.tar.gz>
. Extract tar bzip2 archive	$ tar -xjf <archive.tar.bz2>

Other compressor tools

. Create gzip archive	$ gzip <file>
. Create bzip archive	$ bzip2 -z <file>
. Create zip archive	$ zip archive <file>
. Extract gzip archive	$ gunzip <file.gz>
. Extract bzip2 archive	$ bunzip2 <file.bz2>
. Extract zip archive	$ unzip <file.zip>

Create and edit text files

. create or edit file with vi	$ vi <file-name>
. edit file with vim	$ vim <file-name>

. find files with a specific name	$ find </dir> -name <file-name>
. find files with a specific size	$ find </dir> -size <+100M> -exec ls -l {} \; 2> /dev/null

Create, delete, copy, and move files and directories

. description of the fs hierarchy	$ man hier
. list directories	$ ls -la
. print current directory	$ pwd
. create empty file	$ touch <file-name>
. create a directory	$ mkdir <dir-name>
. create a complete path	$ mkdir -p <path/dir>
. copy files and directories	$ cp <source> <destination>
. move and rename files	$ mv <source> <new-name>
. remove files	$ rm <file-name>
. remove a directory and sub directories	$ rm -fr <dir-name>

Create hard and soft links

. Create hard link	$ ln </file-name> <link-name>
. Create a symbolic link	$ ln -s </file-name> <link-name>
. Link to a directory (symbolic link)	$ ln -s </dir> <link-name>
. Remove a symbolic link	$ unlink <link-name>

List, set, and change standard ugo/rwx permissions

. List ownership and permissions $ ls -la

. change user and group owner on dir and subdirs

$ chown -R <user>:<group> <path/dir>

. change permissions for user, group and others on dir and subdirs

$ chmod -R u=<rwx>,g=<rx>,o=<r> <path/dir>

Locate, read, and use system documentation including man, info, and files in /usr/share/doc

. Install man db and pages $ yum install man-pages man-db man

. Generate man database $ mandb

. open man page $ man <man-page>

. Search specific man page description $ man -k <keyword>

. Search man page for admin utility $ man -k <keyword> | grep 8

Search within a man page
Use **/** then type the search keyword and use **n** to look further within the file

Sections:
(1) User commands, **(5)** Configuration Files, **(7)** Different Topics, **(8)** System Administration

Operate running systems

Boot, reboot, and shut down a system normally

. reboot system	$ shutdown -r now
. shutdown system	$ shutdown -h now

Boot systems into different targets manually

. systemd targets are a group of units	. systemd.unit=emergency.target
that define the state of the system	. systemd.unit=rescue.target
	. systemd.unit=multi-user.target
	. systemd.unit=graphical.target

. show default target	$ systemctl get-default
. set a new default target	$ systemctl set-default <unit.target>
. switching between targets	$ systemctl isolate <unit.target>

Interrupt the boot process in order to gain access to a system (critical skill)

. changing the root password from the boot prompt

+ reboot or start-up the system

+ wait for the grub menu to appear then press [e]

+ find the line starting with 'linux ($root)/vmlinuz' and enter "rd.break" at the end

+ press [Ctrl + x] to boot the system with these options

+ now the root filesystem is mounted in read-only mode to /sysroot

. remount with rw permissions	# mount -o remount,rw /sysroot
. switch /sysroot to "/" filesystem	# chroot /sysroot
. set the new root password	# passwd
. relabel the SELinux contexts !!	# touch /.autorelabel
. restart	# exit
	# exit

Identify CPU/memory intensive processes and kill processes

. show all processes	$ ps -aux
. stop processes	$ kill -15 <PID>
. kill process	$ kill -9 <PID>

. show all procs with a dashboard tool	$ top
	top> press k to select and kill a running process
	top> press f to display more fields

. show all running jobs	$ jobs
. show free ram and swap space	$ free -m
. show average system load	$ uptime
. show amount of CPU`s	$ lscpu

Adjust process scheduling

. run program with modified scheduling prio	$ nice -n <number> <command>
. set process priorities	$ renice -n <number> -p <PID>
. renice with top	$ top
	top> press f to display more fields
	top> press r to renice a process (prioritize)

Locate and interpret system log files and journals

. location containing logs	/var/log
. syslog configuration file	/etc/rsyslog.conf
. journald configuration file	/etc/systemd/journald.conf
. default for message logging	/var /log/messages
. utility to gather all the systemd logging	$ journalctl

Preserve system journals

. Configure systemd to persistently store	$ mkdir /var/log/journal
journal logs	$ systemctl restart systemd-journald.service

Make sure the Storage= parameter in /etc/systemd/journald.conf is set to “auto” or “persistant”

Start, stop, and check the status of network services

. show all services in systemd	$ systemctl --type service
. check state of a specific service	$ systemctl status <service>
. start a specific service	$ systemctl start <service>
. stop a specific service	$ systemctl stop <service>
. enable services to start automatically	$ systemctl enable <service>
. disable services to make sure they won`t start	$ systemctl disable <service>
. show failed services	$ systemctl --failed

Securely transfer files between systems

. send file to remote server

$ scp < source-path /file> username@remote:/<destination-path>

. receive file from remote server

$ scp username@remote:/<source-path/file> <destination-path>

Configure local storage

List, create, delete partitions on MBR and GPT disks

. list current disk devices	$ cat /proc/partitions
. list current disk devices with fdisk	$ fdisk -l
. list block devices	$ lsblk

. creating MBR partition with fdisk (< 2TB disks) Don't forget to write partition to disk with "w" and to push the new partition with partprobe	$ fdisk /dev/<disk "sda">
	fdisk> n (adding a new partition)
	fdisk> w (write partition table to disk)
	$ cat /proc/partitions

. delete MBR partition with fdisk	$ fdisk /dev/<disk>
	fdisk> d (delete partition)
	fdisk> w (write partition table to disk)
	$ cat /proc/partitions

. creating GPT partition with gdisk (> 2TB disks) never use gdisk on a disk with MBR partitions, it will try to convert and possibly break the current MBR partitions	$ yum install gdisk
	$ gdisk /dev/<disk "sdb">
	gdisk> n (adding a new partition)
	gdisk> w (write partition table to disk)
	$ cat /proc/partitions

. delete GPT partition with gdisk	$ gdisk /dev/<disk>
	gdisk> d (delete partition)
	gdisk> w (write partition table to disk)
	$ cat /proc/partitions

. push new partition table to the Kernel	$ partprobe

Create and remove physical volumes

. First create a partition for the Logical Volume make sure to set the partition type to Linux LVM (8e) by pressing "t" in the fdisk menu.	$ fdisk /dev/<disk>
	fdisk> n (adding a new partition)
	fdisk> t (type 8e to give the Linux LVM type)
	fdisk> w (write partition to disk)
	$ partprobe /dev/<disk>

	$ fdisk -l
. list physical volumes	$ pvs
. create a physical volume	$ pvcreate /dev/<partition>
. remove a physical volume	$ pvremove /dev/<partition >

Assign physical volumes to volume groups

. list volume groups	$ vgs
. create a volume group and add physical volume	$ vgcreate <vg-name> <pv-name>
. add physical volume to existing an volume group	$ vgextend <vg-name> <pv-name>
. remove physical volume from volume group	$ vgreduce <vg-name> <pv-name>
. remove an existing volume group	$ vgremove < vg-name>

Create and delete logical volumes

. list logical volumes	$ lvs
. create logical volume with a size of 1GB	$ lvcreate --size 1G --name <lv-name> <vg-name>
. remove logical volume	$ lvremove /dev/<vg-name>/<lv-name>
. extend logical volume	$ lvextend --size <100m> -r /dev/<vg-name>/<lv-name>

Configure systems to mount file systems at boot by universally unique ID (UUID) or label

Mount a partition with UUID

. get the list of disk UUID:	$ blkid
. Edit /etc/fstab file and add this line with the UUID and file system parameters:	*UUID=<uuid> /<dir> <fs_type> defaults 0 0*
. mount the disk or partition:	$ mount /<dir>

Mount a partition with LABEL (recommended)

. set or change the label (ext2, ext3, ext4):	$ e2label /dev/<partition> <label-name>
. set or change the label (xfs):	$ xfs_admin -L <label-name> /dev/<partition>
. list created partition labels	$ blkid
. Edit /etc/fstab file and add this line with the LABEL and file system parameters:	*LABEL=<label> /<dir> <fs_type> defaults 0 0*
. mount partitions	$ mount -a
. validate mount	$ df -H
. disconnect mount	$ umount /dir

Add new partitions and logical volumes, and swap to a system non-destructively

. Adding swap space by creating a new **partition**. After creating a new partition don't forget to give the swap type (82) to the partition

```
$ fdisk /dev/<disk>
  fdisk> n (adding a new partition)
  fdisk> t (type 82 to give the swap type)
  fdisk> w (write partition to disk)
$ partprobe /dev/<disk>
$ mkswap /dev/<partition>
$ swapon /dev/<partition>
$ cat /proc/swaps
```

. Edit the /etc/fstab file and add this line:
(better to replace the beginning of the line with the Label or UUID of the swap partition)

```
/dev/<disk>  swap  swap  defaults  0 0
```

. to remove the swap partition, remove the line in /etc/fstab file and type:

```
$ swapoff /dev/<disk>
```

. Adding swap space by creating a **logical volume** inside a volume group.

```
$ lvcreate --size 1G --name <lv-name> <vg-name>
$ mkswap /dev/<vg-name>/<lv-name>
$ swapon /dev/<vg-name>/<lv-name>
$ cat /proc/swaps
$ ls -l /dev/mapper
```

. list device maper names:

. Edit the /etc/fstab file and add this line:

```
/dev/mapper/<vg-lv>  swap  swap  defaults  0 0
```

. to remove the swap LVM, remove the line in /etc/fstab file and give the following commands

```
$ swapoff /dev/<vg-name>/<lv-name>
$ lvremove /dev/<vg-name>/<lv-name>
```

Create and configure file systems

Create, mount, unmount, and use vfat, ext4, and xfs file systems

. show all current mounts	$ mount
. check all current available filesystems	$ mkfs []tab []tab
. create xfs filesystem on a logical volume	$ mkfs.xfs -L <label> /dev/<vg>/<lv>
. mount filesystem	$ mount /dev/<vg-name>/<lv-name> /<dir>
. mount filesystem permanently by adding this line to /etc/fstab:	/dev/mapper/<vg-lv> /<dir> xfs defaults 0 0
. create ext4 filesystem on a logical volume	$ mkfs.ext4 -L <label> /dev/<vg>/<lv>
. mount filesystem	$ mount /dev/<vg>/<lv> /<dir>
. mount filesystem permanently by adding this line to /etc/fstab:	/dev/mapper/<vg-lv> /<dir> ext4 defaults 0 0
. unmount file system	$ umount /dev/mapper/<vg-lv>
. search for mkfs.vfat package to install	$ yum provides mkfs.vfat

Mount and unmount network file systems using NFS

. show available shares	$ showmount -e <nfs-server>
. mount NFS share permanently by adding this line to /etc/fstab:	$ <FQDN>:/<share> /<dir> nfs _netdev 0 0
. activate configured mount	$ mount -a

autofs for auto mount

. install and enable autofs	$ yum install autofs && systemctl enable autofs
. autofs master configuration file	/etc/auto.master
. new autofs configs:	/etc/auto.<dir_name>
. Configure automount example for /data	$ vi /etc/auto.master
Add the following line to auto.master:	/data /etc/auto.data
Copy existing autofs file for reference:	$ cp /etc/auto.misc </etc/auto.data>
Replace existing line in /etc/auto.data	<subdir> -rw <FQDN>:/data
Restart autofs:	$ systemctl restart autofs
Change directory to /data/files:	$ cd /data/<subdir>

Verify mount: $ mount

Extend existing logical volumes

. show available capacity logical volume $ lvdisplay (-or $ lvs)

. show available capacity volume group $ vgdisplay (-or $ vgs)

. first add diskspace to the volume group $ vgextend <vg-name> /dev/<partition>

. extend the logical volume $ lvextend -L <+100M> -r /dev/<vg>/<lv>

. shrink logical volumes $ lvreduce -L <-100M> -r /dev/<vg>/<lv>

(only works on ext4 never forget to also resize the file system with -r)

Create and configure set-GID directories for collaboration

. create group with GID $ groupadd -g <50000> <group-name>

. change ownership shared dir $ chown nobody:<group-name> /<dir>

. set write permissions + GID (s) and remove permissions for all others: $ chmod g+ws,o-rwx /<dir>

Configure disk compression

. install vdo $ yum install vdo kmod-kvdo

. enable and start vdo $ systemctl enable vdo && systemctl start vdo

. list vdo devices $ vdo list

. monitor running vdo volumes $ vdostats --human-readable

. print vdo configurations $ vdo printConfigFile

. vdo config file (do not edit directly) /etc/vdoconf.yml

. create a vdo device

$ vdo create --name=<vdo-name> --device=<partition> --vdoLogicalSize=5G

. create filesystem on vdo

$ mkfs.xfs -K /dev/mapper/<vdo-name>

. modify vdo settings (example: disable compression)

$ vdo disableCompression --name /dev/mapper/<vdo-name>

. modify vdo settings (example: set non default log file and change bio thread count)

$ vdo modify --name <vdo-name> --logfile /<path>/<file> --vdoBioThreads 2

. mount vdo filesystem by adding the following line to cat/etc/fstab

/dev/mapper/vdo0 /<dir> xfs defaults,_netdev,x-systemd.requires=vdo.service 0 0

Manage layered storage

. install stratis deamon and cli	$ yum install stratisd stratis-cli
. enable and start stratis	$ systemctl enable stratisd && systemctl start stratisd
. list stratis pools	$ stratis pool list
. create stratis pool (first create partitions to add to pool)	$ stratis pool create <pool-name> /dev/<partition> /dev/<partition>
. create stratis pool filesystem	$ stratis filesystem create <pool-name> <fs-name>
. list stratis pool filesystems	$ stratis filesystem list (-and blkid for more info on uuid and mapper)

. mount stratis pool filesystem by adding the following line to cat/etc/fstab

/stratis/<pool-name>/<fs-name> /<dir> xfs defaults 0 0

. extend stratis storage pool (use --help after each flag because bash-completion is not fully working)

$ stratis pool add-data <pool-name> /dev/<partition>

. delete stratis storage pool (always remove filesystems first)

$ stratis filesystem destroy <pool-name> <fs-name>

$ stratis pool destroy <pool-name>

. create stratis filesystem snapshot and mount to directory persistently

$ stratis filesystem snapshot <pool-name> <fs-name> <snapshot-name>
$ vi /etc/fstab (add the following line to fstab)
/stratis/<pool-name>/<snapshot-name> /<dir> xfs defaults 0 0

. rename stratis storage pool and filesystem

$ stratis pool rename <pool-name> <new-pool-name>

$ sratis filesystem rename <pool-name> <fs-name> <new-fs-name>

When not mounted on UUID make sure you edit /etc/fstab with the new names

Diagnose and correct file permission problems

. list ACL (Access Control Lists)	$ getfacl /<dir>
. set ACL for group on dir and sub dirs.	$ setfacl -R -m g:<group>:<rx> /<dir>
. remove permissions allowed to a group	$ setfacl -x g:<user> /<dir_or_file>
. set ACL for user on dir and sub dirs	$ setfacl -R -m u:<user>:<rx> /<dir>
. remove permissions allowed to a user	$ setfacl -x u:<user> /<dir_or_file>

. change user and group owner on dir and subdirs

$ chown -R <user>:<group> <path/dir>

. change permissions for user, group and others on dir and subdirs

$ chmod -R u=<rwx>,g=<rx>,o=<r> <path/dir>

Deploy, configure, and maintain systems

Schedule tasks using at and cron

. use **at** to schedule a job once	$ at
. check if at service is running	$ systemctl status atd
. show the current at que	$ atq
. schedule a job to execute once	$ at <21:30>
	at> <date >> date.file>
	at> []Ctrl + []D

. use **cron** to schedule recurring jobs	$ crontab -e (creates jobs in current users crontab)
. check if at service is running	$ systemctl status crond
. check cron example	$ cat /etc/crontab
. cron configuration files	/etc/crontab (do not modify, managed from rpm`s)
. cron directory for scheduling jobs	/etc/cron.d

. create a cron job that executes every 5	$ vi /etc/cron.d/<jobname>
minutes, add the following line:	*/05 * * * * <user> <command to be executed>

. **anacron** directories for scheduling	/etc/cron.daily
scripts (scripts can be placed into these	/etc/cron.hourly
directories and will be executed	/etc/cron.monthly
according to the dir name)	/etc/cron.weekly

Start and stop services and configure services to start automatically at boot

. show all services in systemd	$ systemctl --type=service
. check state of a specific service	$ systemctl status <service>
. start a specific service	$ systemctl start <service>
. stop a specific service	$ systemctl stop <service>
. enable services to start automatically	$ systemctl enable <service>
. disable services to make sure they won`t start	$ systemctl disable <service>
. show failed services	$ systemctl –failed

Configure systems to boot into a specific target automatically

. show default boot target	$ systemctl get-default
. set default boot target	$ systemctl set-default <*.target>

Configure time service clients

. list time zone information	$ timedatectl status
. list available time zones	$ timedatectl list-timezones
. set timezone	$ timedatectl set-timezone <Europe/Amsterdam>
. install chrony (always disable ntpd)	$ yum install -y chrony
. enable and start chrony service	$ systemctl enable chronyd ; systemctl start chronyd
. chrony configuration file	/etc/chrony.conf
. enable ntp with datetimectl	$ timedatectl set-ntp true

Install and update software packages from Red Hat Network, a remote repository, or from the local file system

. information on yum repositories	$ man yum.conf
. show the current repositories	$ yum repolist
. list the location of the repos	$ ls -l /etc/yum.repos.d/
. search for packages based on name or desc.	$ yum search <keyword>
. do a deep search for packages	$ yum provides */<keyword>
. remove package	$ yum remove <package_name>
. remove package and unneeded dependencies	$ yum autoremove <package_name>
. show all available packages from the repo	$ yum list
. search a specific package from the repo	$ yum list | grep <keyword>
. update packages	$ yum update
. list yum package groups	$ yum groups list hidden | less
. install yum package group	$ yum groups install <package_group_name>
. Check installed package (query rpm database)	$ rpm -qa | grep <package_keyword>
. check all files regarding the installed package	$ rpm -ql <package_name>
. check from what package a file is coming	$ rpm -qf /<dir>/<file>
. location for binaries outside of the normal distribution:	/usr/local/bin

Work with package module streams

. list all modules	$ yum module list
. list installed modules	$ yum module list installed

. find which module provides a package	$ yum module provides <package>
. display the current status of a module	$ yum module list <module>
. examine details of a module	$ yum module info <module>
. enable a stream without installing packages	$ yum module enable <module:stream>
. install a specific stream	$ yum module install <module:stream/profile>
. disable a module stream and remove all packages provided by it	$ yum module remove module && yum module disable module

Modify the system bootloader

. GRUB2 configuration file	/etc/default/grub
. Push changes to the GRUB2 config (run after changes are made in /etc/default/grub)	$ grub2-mkconfig -o /boot/grub2/grub.cfg

Manage basic networking

Configure IPv4 and IPv6 addresses

. network connection configuration files	/etc/sysconfig/network-scripts/ifcfg-<conn-name>
. list nic-adapter configuration	$ ip address show (-or $ nmcli device show)
. list routing tables	$ ip route show
. list adapters status and connections	$ nmcli connection show (-or $ nmcli device status)
. start network connection	$ nmcli connection up <conn-name>
. stop network connection	$ nmcli connection down <conn-name>
. restart network manager	$ systemctl restart NetworkManager.service
. remove a specific connection	$ nmcli connection del <conn-name>

. configure ipv4 ip-adress, subnetmask and gateway for a specific connection

$ nmcli connection add ifname <eth0> type ethernet ipv4.adresses <ip/netmask> ipv4.gateway <gw-ip>

. configure ipv6 ip-adress, subnetmask and dns for an existing connection

$ nmcli connection modify <conn-name> ipv6.addresses <ip/netmask> ipv6.dns <dns-server-ip>

. add persistant route with nmcli

$ nmcli connection modify <conn-name> ipv4.routes "<ip/netmask> <gateway>"

. troubleshoot dhcp	$ dhclient
. graphical network management tool	$ nmtui (nmcli is preferred over graphical utility)

Configure hostname resolution

. hostname configuration file	/etc/hostname
. list hostname	$ hostnamectl (-or $ hostname)
. set or modify hostname	$ hostnamectl set-hostname <fqdn>
. local resolvable hostnames	/etc/hosts
. dns configuration file	/etc/resolv.conf

. to modify or add the dns server address add this line to /etc/resolv.conf

nameserver <IP>

. use nmcli to add DNS server IP to an existing connection

$ nmcli connection modify <conn-name> ipv4.dns <dns-server-ip>

Configure network services to start automatically at boot

. start network service at boot	$ systemctl enable <network_service>

. prevent network service to start at boot $ systemctl disable <network_service>

Restrict network access using firewall-cmd/firewall

. enable and start firewalld $ systemctl enable firewalld && systemctl start firewalld

. disable firewalld enabled services $ firewall-cmd --remove-service <name> --permanent

. block firewalld enabled ports $ firewall-cmd --remove-port <port>/<protocol> --permanent

. add rich rule $ firewall-cmd --add-rich-rule=”<rich_rule>” --permanent

. Examples to add rich rules $ man firewalld.richlanguage

. go to examples paragraph: /EXAMPLES

. restrict ssh access coming from nodes in the 192.168.122.0/24 network

```
$ firewall-cmd --add-rich-rule=”rule family= ipv4 source address= 192.168.122.0/24 \
  service name= ssh accept” --permanent
$ firewall-cmd --remove-service ssh --permanent
$ firewall-cmd --reload
$ firewall-cmd --list-all
```

Manage users and groups

Create, delete, and modify local user accounts

. show user properties and group memberships	$ id <username>
. create a local user account	$ useradd <user>
. remove a local user and home dir (r)	$ userdel -r <user>
. change the name of a user account	$ usermod -l <new-name> <user>
. add user to group	$ usermod -aG <group> <user>

Change passwords and adjust password aging for local user accounts

. assign a password to a local user account	$ passwd <user>
. set password expiration date for local user	$ chage -E <YYYY-MM-DD> <user>
. show local user password expiry information	$ chage -l <user>

Create, delete, and modify local groups and group memberships

. create a group without any associated user	$ groupadd <group>
. create a group without any associated user	$ groupadd -g <GID> <group>
. change the name of a group	$ groupmod -n <new-name> <group>
. remove a group without any associated user	$ groupdel <group>
. change the GID of a group	$ groupmod -g <GID> <group>
. add a secondary group to a user account	$ gpasswd -a <user> <group>
. to remove a user from a secondary group	$ gpasswd -d <user> <group>
. get the list of the members of a given group	$ groupmems -g <group> -l

Configure superuser access

. sudo command policy config	/etc/sudoers
. make sure this line is enabled:	%wheel ALL=(ALL) ALL
. Add the user to the wheel group	usermod -aG wheel <user-name>

Manage security

Configure firewall settings using firewall-cmd/firewalld

. check firewall service	$ systemctl status firewalld
. show applied firewalld componentes	$ firewall-cmd --list-all
. view all available services	$ firewall-cmd --get-services
. view all applied services	$ firewall-cmd --list-services
. location of firewalld services	/usr/lib/firewalld/services
. Path for custom created services	/etc/firewalld/services

. creating a new service (by copy an existing service and apply it)	$ cp /usr/lib/firewalld/services/ssh.xml /etc/firewalld/services/<name>.xml
	$ vim /etc/firewalld/services/<name>.xml
	(Edit the name, descry, ports and protocols)
	$ firewall-cmd --reload
	$ firewall-cmd --add-service <name> --permanent
	$ firewall-cmd --reload
	$ firewall-cmd --list-all

. add custom ports	$ firewall-cmd --add-port=<port>/<protocol> --permanent
. List all rules with IP Tables	$ iptables -L

Configure key-based authentication for SSH

. Use SSH keys	$ ssh-keygen
. use the defaults, no passphrase	[]enter
. distribute the public key (id_rsa.pub)	$ ssh-copy-id <hostname_or_ip>

Set enforcing and permissive modes for SELinux

. current configuration	/etc/sysconfig/selinux
. get selinux status	$ sestatus
. check current mode	$ getenforce
. when enforced how to switch to permissive	$ setenforce Permissive
. disable selinux and make options persistant:	$ vim /etc/sysconfig/selinux
. set the following option in /etc/sysconfig/selinux:	SELINUX=disabled
. reboot the system:	$ shutdown -r now

List and identify SELinux file and process context

. get a SELinux file context, type	$ ls -Z
. get a SELinux process context, type	$ ps -eZ
. how to view context types of ports	$ netstat -Ztulpen

Restore default file contexts

. restore the default SELinux file contexts	$ restorecon -Rv </path>

Use boolean settings to modify system SELinux settings

. show list of SELinux booleans	$ getsebool -a
. enable specific boolean persistant	$ setsebool -P <boolean_name> on
. disable specific boolean persistant	$ setsebool -P <boolean_name> off
. get detailed list of SELinux booleans	$ yum install setroubleshoot-server
	$ semanage boolean -l

Diagnose and address routine SELinux policy violations

. install setroubleshoot-server package	$ yum install setroubleshoot-server
. display the SELinux policy violations	$ sealert -a /var/log/audit/audit.log
. look for sealert messages	$ less /var/log/messages | grep sealert
. more info	$ less /var/log/audit/audit.log | grep AVC

EXTRA

List and add persistant routes

. persistant routes configuration files	/etc/sysconfig/network-scripts/route-<eth0>
. list router and configured routes	$ ip route show (-or $ route -n)
. add temp route	$ ip route add <IP/SN> via <Gateway> dev <eth0>
. to add a persistant route create a file named /etc/sysconfig/network-scripts/route-<*eth0*> and add this line to the file:	<ip/netmask> via <gw-ip> dev <eth0>

Access a virtual machine's console

. Install all KVM software packages	$ yum install -y kvm libvirt virt-manager qemu-kvm
. Access console in x-windows	$ virt-manager
. with KVM from the console	$ virsh console <vm_name>

Start and stop virtual machines

. show virtual machines with virsh	$ virsh list –al
. start virtual machine with virsh	$ virsh start <vm_name>
. stop virtual machine with virsh	$ virsh shutdown <vm_name>
. reboot virtual machine with virsh	$ virsh reboot <vm_name>
. start graphical vm manager	$ virt-manager

Configure systems to launch virtual machines at boot

. get a list of all virtual machines	$ virsh list --all
. launch virtual machine at boot	$ virsh autostart <vm_name>
. disable launch at boot	$ virsh autostart <vm_name> --disable

Update the kernel package appropriately to ensure a bootable system

. install a new kernel package	$ yum install kernel
. update kernel (old kernel is not being removed, stays available in the bootloader)	$ yum update kernel

More useful utilities and commands for troubleshooting network related issues

. install bind-utils (dns client) package	$ yum provides nslookup
. resolve dns names	$ nslookup <IP> (-or $ dig <IP>)
. send ip packets to remote hosts	$ ping <IP>
. measure the network reliability (flood)	$ ping -f <IP>

. measure the network reliability with specific packet size

$ ping -s <packet_size_in_bytes> -f <IP>

. Show all routers/gateways (devices) between the host and the destination:

$ traceroute <IP>

. show and analyse ports that are listening on the system with socket statistics (preferred utillity)

$ ss -tuna | grep <port>

. show and analyse ports that are listening on the system with netstat (depcricated utillity)

$ sudo netstat -ano | grep <port>

. List the TCP ports that are being listened on, and the name of each listener's daemon and its PID

$ sudo netstat -tulpen

. test if remote port is listening with netcat, returns 0 when successfully connected

$ nc -z <FQDN/IP> <port>; echo $?

. Configure netcat in listener mode

$ nc -l <port> -e /bin/bash

. nmap port scanner to see what ports are listening on a remote hosts

$ nmap <FQDN/IP-address>

. scan network range

$ nmap -n <network/mask>